Key Agreement Protocols

Jong Hwan Park

Authentication and Session Key

1. PKE
2. DS + DH
(g, p)

- Usually, a **session key** is required
  - I.e., a symmetric-key for a particular session
  - Then, used for confidentiality (using SKE) and/or integrity (using MAC)
- How to authenticate and establish a session key (shared symmetric-key)?
  - Need key agreement protocols
  - In practice, authentication and key agreement are done simultaneously
  - During a secure protocol for two purposes
    - Attacker cannot break the authentication…
    - …and attacker cannot determine the session key
- Need crypt primitives
  - Diffie-Hellman, PKE, DSS (especially for establishing a session key)
  - SKE, MAC, hash,…

Authentication & Key Agreement

- Possible cases:
  - Depending on unilateral or mutual authentication
  - Depending on authentication-only or both purposes

Protocols	Authentication	Key Agreement
P-1	Unilateral	X
P-2	Mutual	X
P-3	Unilateral	O
P-4	Mutual	O

상황에따라 중요.

  - Each protocols are designed:
    - By using various crypto primitives (symmetric, asymmetric)
    - Under assumption that relates to Alice and Bob (key storage, state)

가정이 중요하다.

Using DH+DS+SKE (1)

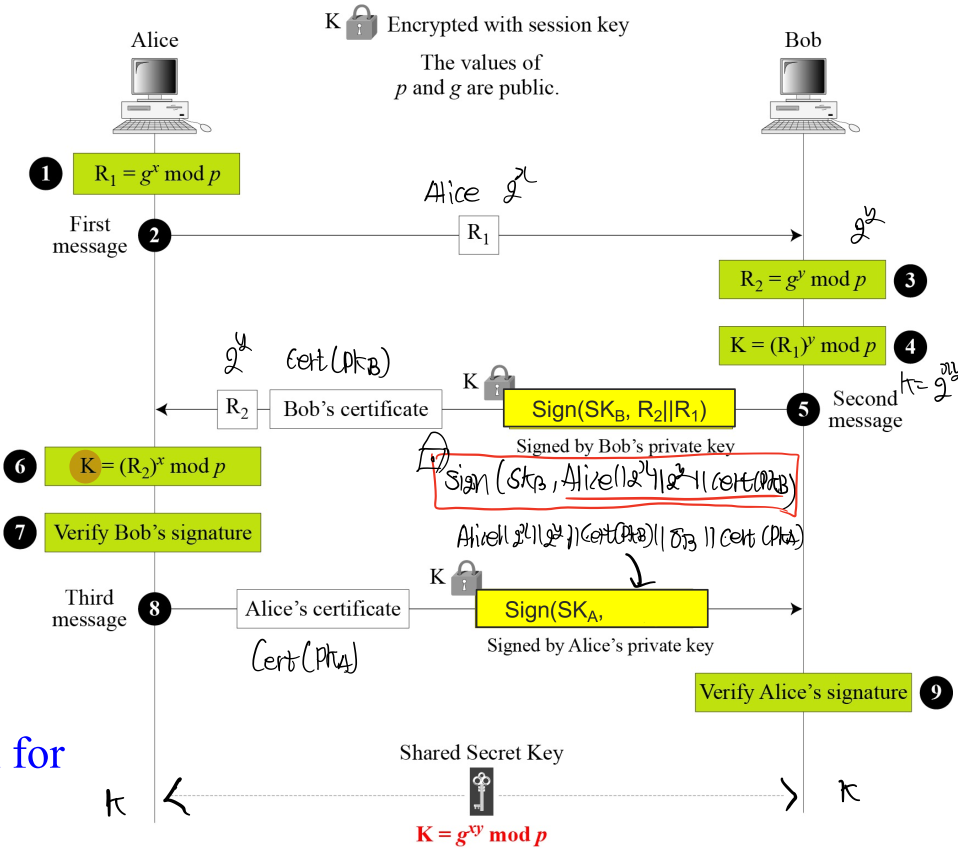
- Authenticated Diffie-Hellman key-exchange protocol
  - (a.k.a) Station-to-Station (STS) protocol

$Cert_A = Sign_{CA}(Alice, PK_A)$

$Cert_B = Sign_{CA}(Bob, PK_B)$

둘다 전자서명을 공유한다.

DH (g . p) 가 공유되었다 가정



- What is the assumption for Alice and Bob?

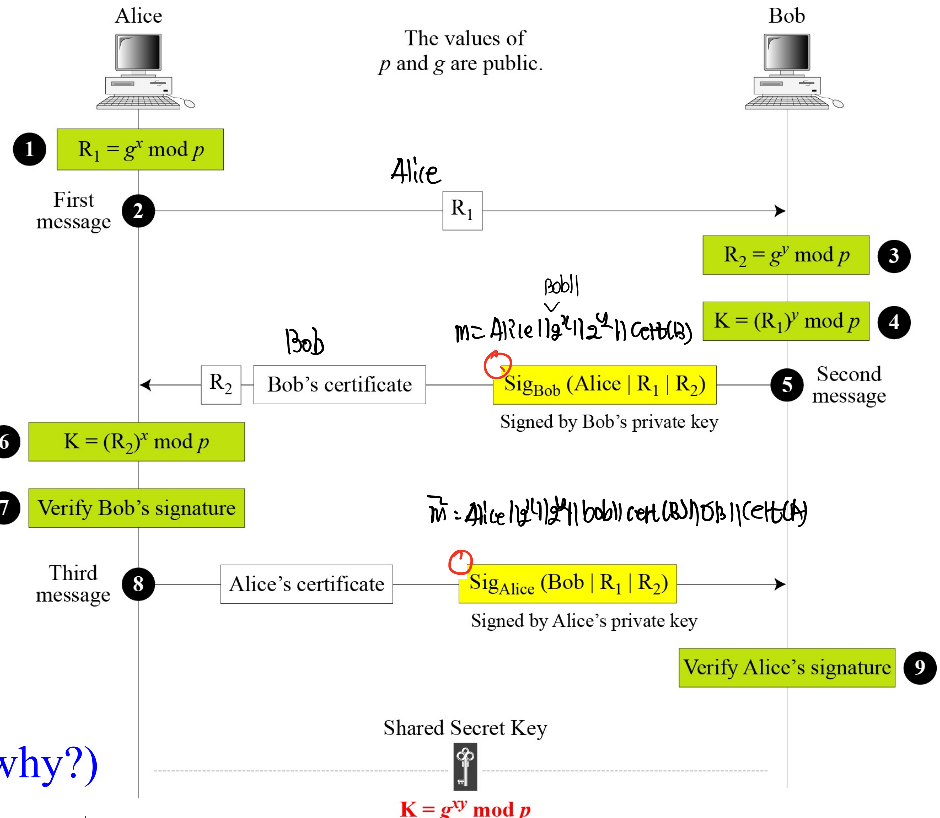
Using DH+DS (2)

- Simplified STS protocol
  - What are differences?

TLS V1.2 → TLS V1.3

세번째부터 암호화.

· 두사람이 cert를 가진다.



  - Mutual authentication (why?)
  - Authenticated key agreement

Encrypted Key Exchange (EKE)

- PW-based authentication suffers from offline dictionary attack
- EKE protocol (= PW + PKE + SKE)　　Alice $\xrightarrow{h(pw)}$ Bob
  - Assume Alice and Bob share a password PW
  - Goal is to gain mutual authentication and a shared key

Attack
① - - -
② - - -
PK_1
PK_2
⋮
PK_n

KeyGen
→ (PK, SK)

Alice → Bob: Alice, $SKE_{PW}(PK)$　　A random PK

없애면? 의미는 없다.

Bob → Alice: $SKE_{PW}(E_{PK}(K))$　　A random K

Alice → Bob: $SKE_K(R_A)$　　challenge

Bob → Alice: $SKE_K(R_A, R_B)$

Alice → Bob: $SKE_K(R_B)$

Alice　PW
(Client)

Bob　[Alice, PW]
(Server)

초기는 cert가 필요.　Cert(PK_B)
↳ PKI 필요없다.

- Why can offline dictionary attack be prevented?
- How can it be realized using Diffie-Hellman key agreement? (see later)

EKE based on Diffie-Hellman

- EKE protocol (=PW+DH+SKE)
  - Assume Alice and Bob share a password PW
  - Assume (p, g) are public
  - Goal is to gain mutual authentication and a shared key

Alice PW

(Client)

$K = g^{xy}$

Alice, $SKE_{PW}(g^x)$ → A random x

← $SKE_{PW}(g^y)$ A random y

$SKE_K(R_A)$ →

← $SKE_K(R_A, R_B)$

$SKE_K(R_B)$ →

Bob [Alice, PW]

(Server)

$K = g^{xy}$

  - Still, can prevent offline dictionary attack

Using PKE(1)

- Assume Alice and Bob have PK_A and PK_B for (only) encryption
- A protocol example:

challenge

$E(PK_B, Alice \| K_{AB} \| R_A)$

response.

challenge

$E(PK_A, Bob \| K_{BA} \| R_A \| R_B)$

R_B 보냄

MAC(K, all above messages)

$m = C_A \| C_B \| R_A \| R_B$

Alice (PK$_A$, SK$_A$)

k

Bob (PK$_B$, SK$_B$)

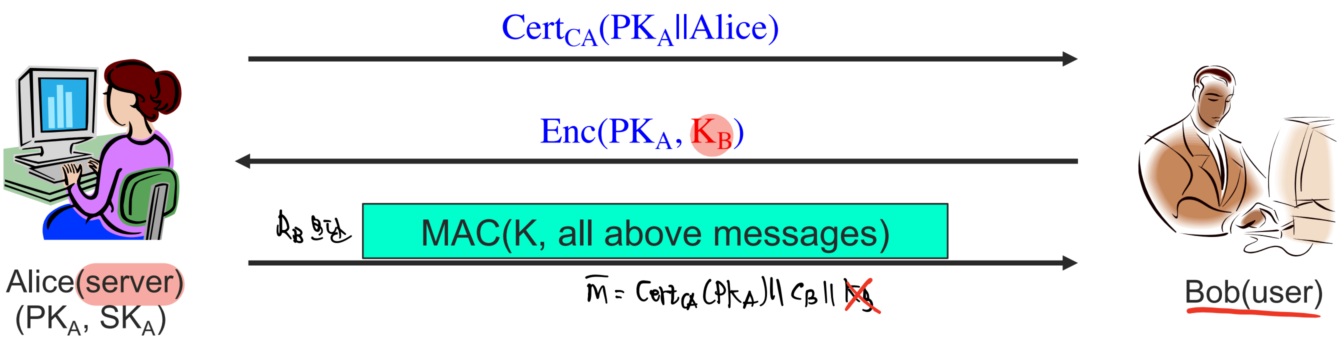
k

  - Mutual authentication
  - K_{AB} and K_{BA} may be combined using hash to form $K=h(K_{AB}, K_{BA})$

세션 키의 결합

Using PKE(2) 현실에서 사용

- <u>Assume Alice has PK_A for (only) encryption</u>
- A protocol example:



암호화된 채널에서 사용자인증 (PW/OTP/cert(PK_A)/Bio/OT-CR (onetime-challenge response)
(SMS, ARS, email))

- authentication Bob이 Alice를 인증 (Client가 web server를 인증)
- K may be computed by hashing $K=h(K_B)$
- In TLS v1.2, RSA encryption is now used

Using DS (?)

- Assume Alice and Bob have PK_A and PK_B for (only) signature
- A protocol example:

서명검증시 메시지가 노출되어야 검증이 가능하다.
= 숨기는것이 없다.

Alice (PK_A, SK_A)

Alice → Bob: Alice, R_A

Bob → Alice: R_B, Sign(SK_B, $R_A \| R_B \|$ Alice $\| K$)   Random

Alice → Bob: Sign(SK_A, $R_B \| R_A \|$ Bob $\| K$)

Bob (PK_B, SK_B)

K

- Mutual authentication is good
- But, session key K is not secret (why?) 인증은 가능하나 세션 key 교환은 X.
  - To establish a session key, need PKE or Diffie-Hellman key exchange

DS+DH

Using DS+DH(1)

- <u>Assume</u>
  - Alice and Bob have (PK_A, SK_A) and (PK_B, SK_B) for DSS, resp.
  - DH parameter is shared as (g, p)

- Auth. & key exchange protocol

Alice (PK_A, SK_A)

Alice, g^x →

← g^y, Sign(**SK**$_B$, $\underbrace{\text{Alice}||g^x||g^y}_{}$), Bob, Cert($PK_B$)

σ_A

Sign(**SK**$_A$, Alice$||g^x||g^y||\sigma_A||$Bob$||$Cert(PK_B)), Cert(PK_A) →

Bob (PK_B, SK_B)

  - Mutual authentication seems to be OK
  - K may be combined using hash to form $K=h(g^{xy})$ HKDF(g^{xy}) = k

Using DS+DH(2) 현실에서 사용

- Assume
  - Bob has (PK_B, SK_B) for DSS
  - DH parameter is shared as (g, p)

Alice가 Bob을 인증 (user가 server를 인증)

- One-way auth. & key exchange protocol

Alice, g^x →

← g^y, Sign(SK_B, Alice$\|g^x\|g^y$), Bob, Cert(PK_B)

MAC(K, Alice$\|g^x\|g^y\|\sigma_B\|$Bob$\|$Cert(PK_B)) →

암호화된 채널에서 사용자인증 (PW/OTP/cert(PK_A)/Bio/OT-CR (onetime-challenge response)
(SMS, ARS, email)

  - K may be combined using hash to form K=h(g^{xy})
  - In TLS v1.2, ECDSA is used for DSS and ECDH is used for DH

서명

Simplified TLS Protocol (1)

Encryption

- Assume only server has (PK_B, SK_B) and cert(PK_B)

암호들의 조합 (암호학적 primitive)

서버가 ciphersuite를 하나 선택.

Alice, ciphersuites, R_A →

Random number.

← Cert(PK_B), cipher, R_B

random

E(PK_B, S), MAC(K, h(msgs)) ① →

← MAC(K, h(msgs)) ②

Alice

Web Server / Bob (PK_B, SK_B)

Data protected with key K

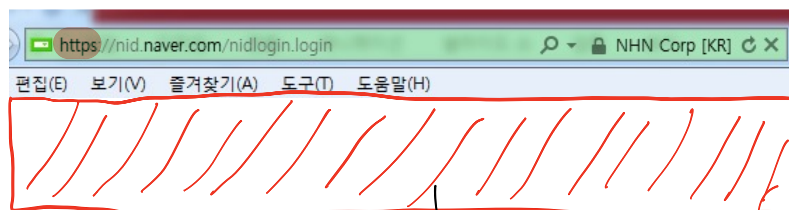
K K

①에서 MAC 검증을 하는 이유

서버에서 사용자와 동일한 K를 공유했는지 알고싶어한다.

②에서

Alice 역시 K를 Server가 공유한 것을 알 수 있다.



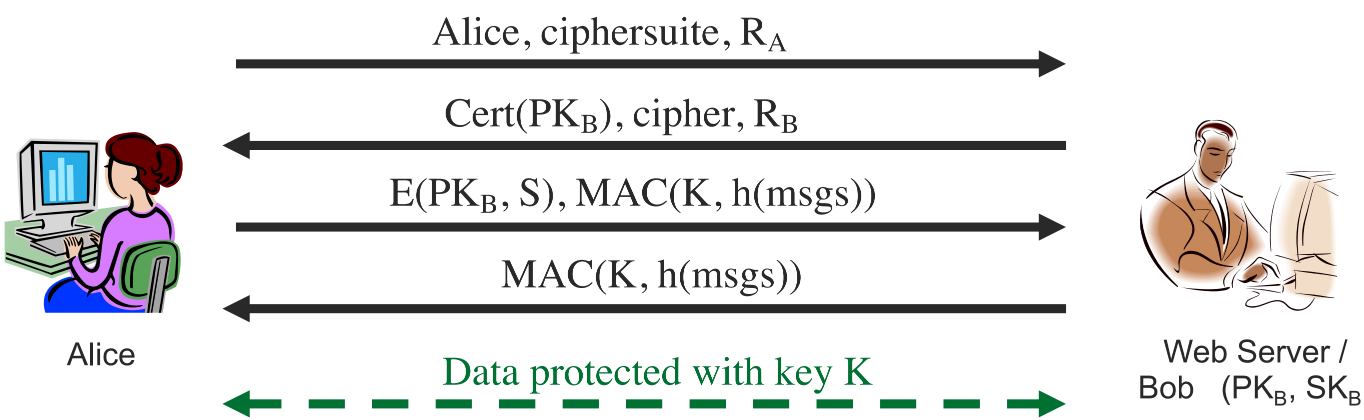
암호화 되어있다.

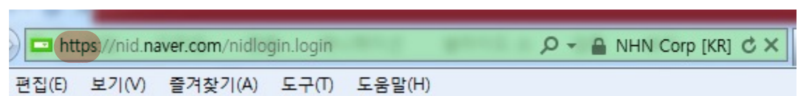
- Note:
  - Unilateral authentication (why in practice ?)
  - S is known as **pre-master secret**
  - K = h(S, R_A, R_B) and "msgs" means all previous messages

Simplified TLS Protocol (2)

Encryption

- Assume only server has (PK_B, SK_B) and cert(PK_B)





이것이 가능하면 Pishing이라는 서버를 사칭하는것이 불가능하다.

- Is Alice sure she is talking to Bob(web server)?
- Is Bob sure he is talking to Alice(Client)?
  - What if Bob also wants to authenticate Alice?
  - Can either request cert(PK_A) or password in the second message transit

Simplified TLS Protocol (3)

Signature

- Assume only server has (PK_B, SK_B) and cert(PK_B)

"DS + DH"

Alice

Alice, ciphersuites, R_A →

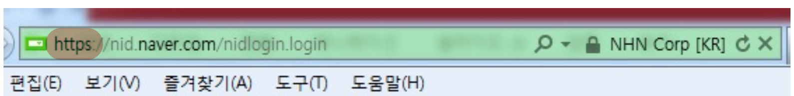
← Cert(PK_B), cipher, R_B

g^x →

← g^y, Sign(SK_B, h(msgs))

← Data protected with key K →

Web Server / Bob (PK_B, SK_B)

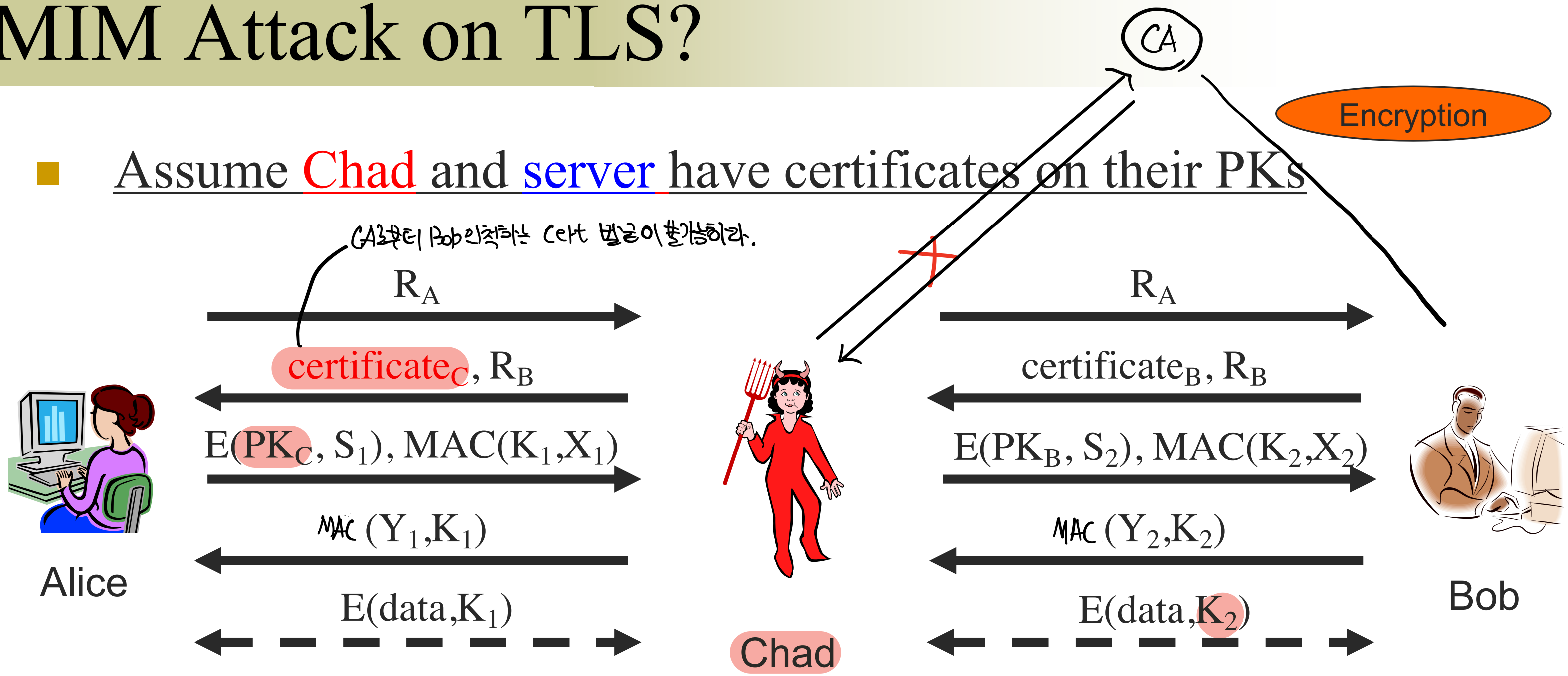


- Note: ·이역시 Alice 만이 Bob을 인증한다.
  - Unilateral authentication (why in practice ?)
  - $K = h(g^{xy}, R_A, R_B)$ and "msgs" means all previous messages
  - Can either request cert(PK_A) or password in the second message transit

MIM Attack on TLS?

- Assume Chad and server have certificates on their PKs

CA로부터 Bob인척하는 cert 발급이 불가능하다.

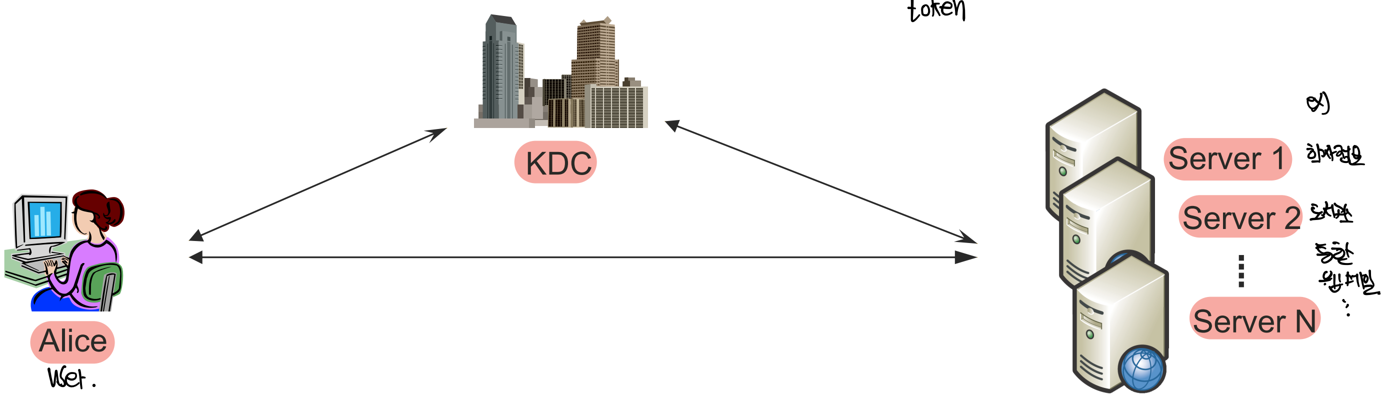


- What prevents this MIM attack?
- Bob's certificate must be signed by a certificate authority (CA)
  - What does browser do if signature not valid?
  - What does user do when browser complains?

Kerberos Protocol

- Authentication and key agreement protocol
  - devised by MIT, and now realized in Windows 2000
  - Run in symmetric-key setting (No PK setup!)
  - KDC (Key Distribution Center) is required – three-party
  - Environments for Kerberos
    - When N servers give their services to a user, it would be unrealistic to require a user to memorize N passwords (or hold N smartcards)
    - E.g., Samsung business units = {Electronics, Auto, Cards, Insurance, …}
  - Enable the user to maintain one password (Single-Sign-On) SSO

"token"

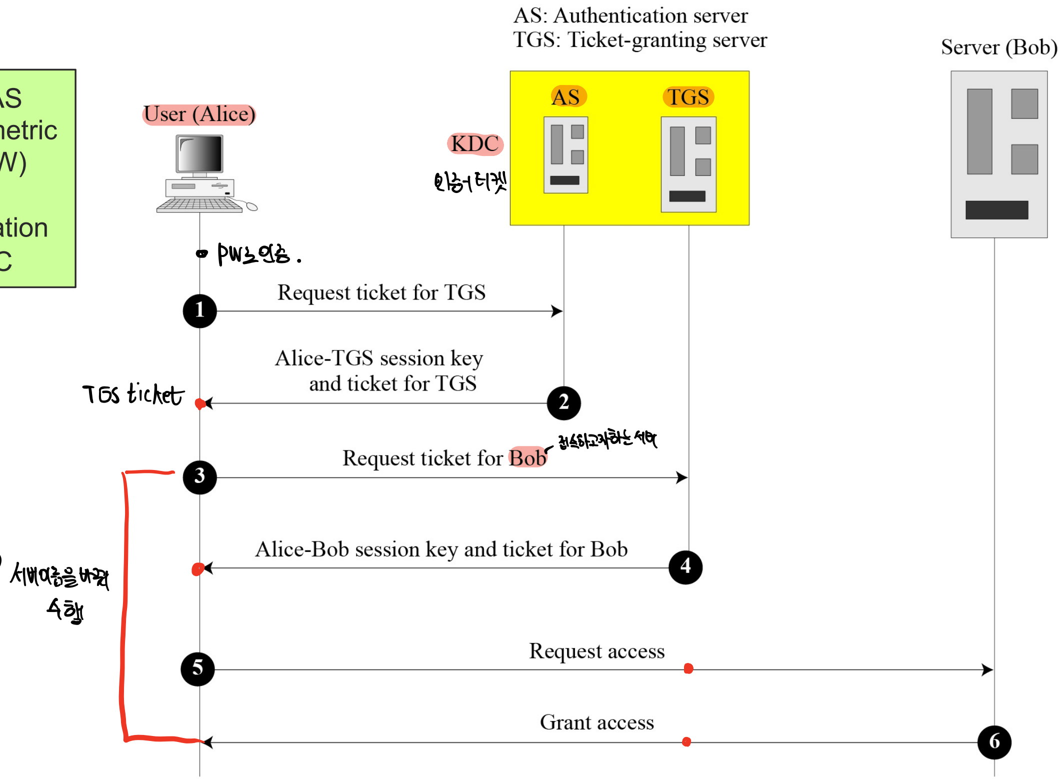


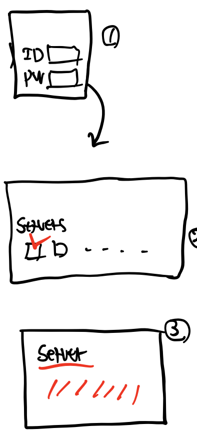
Kerberos Architecture

- KDC consists of two authentication servers
  - AS (Authentication Server) and TGS (Ticket Granting Server)
- Assume a user and AS share a password (PW)

User and AS share a symmetric key K=h(PW)

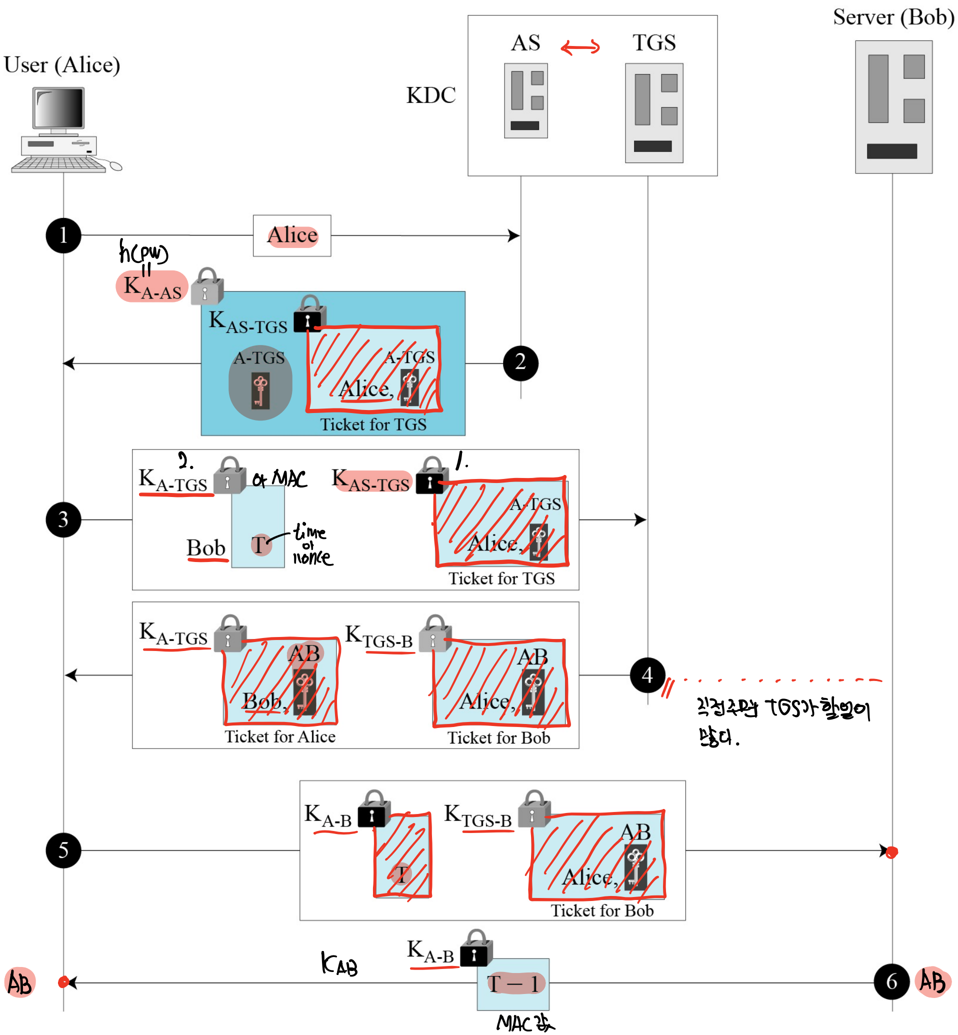
→ Authentication using MAC





If Alice needs to receive services from different servers, repeat (3)~(6) steps

How Kerberos works



K_{A-AS} Encrypted with Alice-AS key

K_{TGS-B} Encrypted with TGS-Bob key

K_{A-TGS} Encrypted with AS-TGS key

K_{AS-TGS} Encrypted with Alice-TGS session key

K_{A-B} Encrypted with Alice-Bob session key

A-TGS Alice-TGS session key

AB Alice-Bob session key

KDC: Key-distribution center

AS: Authentication server

TGS: Ticket-granting server

T: Timestamp (nonce)

Authenticator in (3) and (5)
→ Using MAC is more desirable
T값은 이미 노출되어 있으므로

What if the ticket for Bob in (4) is sent to Bob directly?

→ Stateless KDC

Q & A